OpenMP: Cercador d'imatges

Data d'entrega: 17 de maig del 2015

Índex

1 Introducció

L'objectiu d'aquesta pràctica és implementar una aplicació que permeti cercar imatges fent servir l'histograma per descriure-les. En particular, l'aplicació ha d'implementar les següents funcionalitats

1. Extracció de l'histograma de les imatges.
2. Cerca d'imatges similars a partir d'una imatge exemple.
3. Persistència en les dades extretes de la imatge.

Cadascuna de les funcionalitats anteriors es detallarà a les següent seccions.

2 Funcionalitats

Aquesta secció detalla les funcionalitats a implementar per l'aplicació.

2.1 Construcció de la base de dades

Per construir la base de dades d'histogrames de les imatges es proposa procedir de la següent forma per a construir-la.

- L'aplicació ha de permetre seleccionar un fitxer de text pla que conté els noms de fitxer de les imatges que s'han d'afegir a la base de dades. Cada imatge estarà especificat en una línia separada i poden estar emmagatzemades en qualsevol directori del disc. Podeu suposar que no es coneix el nombre d'imatges que hi ha en aquest fitxer.
- Cada imatge es copiarà al directori `imatges` de l'aplicació. Les imatges es guardaran amb el nom `img_XXXXXX.jpg` on `XXXXXX` és un número sencer, anomenat *identificador*, que s'anirà incrementant de forma automàtica a mesura que s'afegeixen imatges a la base de dades. Així, la primera imatge es guardarà amb el nom `hist_000001.jpg`, el segon amb `img_000002.jpg`, i així successivament.
- Per a cada imatge es construirà l'histograma corresponent. Els noms de fitxer dels histogrames es guardaran al directori `histogrames` amb el nom `hist_XXXXXX.xml`, on `XXXXXX` és un número sencer que correspon a l'identificador assignat a la imatge. El contingut d'aquest fitxer

XML es detalla a la Secció 4. A més de guardar els fitxers d'histograma a disc (per la persistència), es recomana emmagatzemar també els histogrames a memòria RAM. D'aquesta forma serà més ràpid accedir-hi a l'hora de fer una cerca. Observar que les imatges no es guarden a memòria.

- El primer cop que l'usuari afegeixi imatges a la base de dades es començarà amb un *identificador* igual a 1. En cas que ja existeixin imatges a la base de dades, caldrà inicialitzar l'*identificador* al primer sencer que no s'ha fet servir.

2.2 Cerca d'una imatge en una base de dades

Un cop construïda la base de dades l'objectiu és desenvolupar una eina que permeti realitzar cerques d'imatges. Per això l'aplicació permetrà escollir a l'usuari una imatge qualsevol (que no té perquè pertànyer a la base de dades). L'aplicació construirà l'histograma d'aquesta imatge i el compararà amb tots els histogrames que hi ha a la base de dades. Un cop realitzada aquesta comparació caldrà ordenar els resultats de comparació i presentar-los per pantalla.

2.3 Persistència

Una funcionalitat interessant de l'aplicació és la persistència. Recordar que els histogrames de les imatges indexades es guarden a disc. Per tant, en iniciar l'aplicació (o la primera vegada que es realitza una cerca) es carregaran tots els histogrames de la base dades a memòria RAM.

3 Implementació del paral·lelisme

Es recomana escriure primer una aplicació seqüencial (no paral·lela) que implementi la funcionalitat demana. No paral·lelitzeu fins que no estigueu segurs que l'aplicació funciona correctament. Tingueu en compte que paral·lelitzar amb OpenMP significa inserir les directives de compilador necessaris en el codi. Aquest procés serà relativament senzill sempre i quan el codi estigui ben estructurat d'acord amb les regles d'OpenMP (per exemple, eviteu l'ús de la instrucció "breaks").

A continuació es detallen aquelles parts de l'aplicació que s'han de paral·lelitzar:

3.1 Construcció de la base de dades

En afegir histogrames a la base de dades es realitzarà un paral·lelisme a nivell d'imatge. Tal com s'ha comentat abans, s'especificarà mitjançant un fitxer de text pla el conjunt d'imatges de qual volem extreure els histogrames. A cada línia hi haurà indicat una única imatge i podeu suposar que no se sap amb antelació el nombre d'imatges que hi ha en aquest fitxer. Per tant, cadascuna d'aquestes línies està associada a una tasca. La tasca està composada per la lectura de la imatge de disc, la construcció de l'histograma, l'emmagatzematge de la imatge i l'histograma a disc (veure Secció 2.1) i l'emmagatzematge de l'histograma a memòria RAM. Useu l'exemple del directori `histextract` per veure un exemple de com construir l'histograma i guardar-lo a disc. Observeu que la construcció de l'histograma es realitza per a cada canal de color de forma independent. Es podrà utilitzar doncs també paral·lelisme per construir l'histograma.

Es proposa utilitzar l'espai de color HSV (Hue-Saturation-Value) per a extreure l'histograma de les imatges. Aquest espai de color és un espai en què se separen les components de crominància (hue i saturation) de la de luminància (value). L'espai de color HSV modela millor la visió humana que altres espais de color, com per exemple, RGB.

3.2 Cerca d'una imatge en una base de dades

Per realitzar una cerca també es pot ús de paral·lelisme. L'usuari proveeix a l'aplicació amb un imatge a cercar. Primer es construeix l'histograma d'aquesta imatge fent servir el mateix algorisme que heu fet servir per construir l'histograma de les imatges de la base de dades. A continuació es compara aquest histograma amb totes les imatges que hi ha a la base de dades. La comparació es realitzarà amb els histogrames guardats memòria RAM, ja que llegir les dades de disc no permetrà obtenir la mateixa acceleració que accedir-hi directament a memòria (algú ho vol comprovar ?). El procés de comparació d'histogrames és altament paral·lelizable ja que múltiples fils realitzen la mesura de comparació amb diferents imatges a la vegada. A l'hora de realitzar la comparació, podeu suposar que es coneix el nombre d'imatges que conté la base de dades. Un cop finalitzada la comparació s'ordenaran (de més

semblant a menys semblant) les imatges. Només caldrà presentar a pantalla els millor candidats (per exemple, 15 primers). Al directori `histcompare` trobareu un exemple que permet comparar dos histogrames qualssevol llegits de disc.

3.3 Persistència

En iniciar l'aplicació aquesta haurà de mirar si hi ha alguna imatge a la base de dades. Una forma senzilla de saber-ho és guardar a disc el darrer *identificador* (veure secció 2.1) utilitzat. Si aquest no es troba a disc o bé està inicialitzat a 0, podeu suposar que no hi ha cap imatge a la base de dades. En cas contrari, aquest número us indica el nombre d'imatges que hi ha a la base de dades. Caldrà llegir tots els histogrames i emmagatzemar-los a memòria RAM. Observeu que el procés de lectura es pot realitzar en paral·lel.

3.4 Construcció en la tasca o el bucle ?

Tingueu en compte que per paral·lelitzar un algorisme amb OpenMP podem fer servir, entre altres, la construcció en la tasca i la construcció en el bucle. La construcció en en bucle és més recomanable que la construcció en la tasca ja que l'*overhead* associat és molt menor. Recordeu però que la construcció en en bucle es pot fer sevir només si es coneix prèviament el nombre d'elements a processar. En cas contrari, s'ha d'utilitzar la construcció en la tasca.

3.5 Compilació de codi OpenMP

Per compilar codi OpenMP només cal afegir les següents línies al fitxer `CMakeLists.txt`:

```
FIND_PACKAGE(OpenMP REQUIRED)
if (OPENMP_FOUND)
  set (CMAKE_C_FLAGS "${CMAKE_C_FLAGS} ${OpenMP_C_FLAGS}")
  set (CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} ${OpenMP_CXX_FLAGS}")
endif()
```

4 Estructura del fitxer XML de l'histograma

Els exemples `histextract` i `histcompare` disponibles en aquest directori mostren com es pot guardar i llegir un histograma en format XML. Tal com es pot veure, mitjançant OpenCV és un procés ben senzill.

A la Figura 1 es mostra un exemple de l'estructura XML que es pot fer servir. El fitxer conté la següent informació:

- Un camp `imageName` en què s'emmagatzema el nom de fitxer associat a l'histograma (en el context de la pràctica realment no és necessari aquest camp, ja que a partir del nom de fitxer d'histograma es pot extreure el nom de fitxer de la imatge).
- Tres camps amb els histogrames de la component H (`hist_h`), S (`hist_s`) i V (`hist_v`) respectivament.

5 Què s'ha d'entregar ?

En aquesta secció es detalla el que s'ha d'entregar:

- El fitxer que entregueu s'ha d'anomenar `P1_Cognom1Cognom2.tar.gz` (o .zip, o .rar, etc). `Cognom1` correspon al primer cognom del primer membre del grup, mentre que `Cognom2` correspon al primer cognom del segon membre del grup. Dintre d'aquest fitxer hi ha d'haver tres carpetes: `src`, que contindrà el codi font, `exe`, que contindrà el codi executable, i `doc`, que contindrà un petit document explicatiu.
- La carpeta `src` contindrà tot el codi font. Aquesta carpeta ha d'incloure tots els fitxers necessaris per compilar i generar l'executable a les màquines de l'aula IA. Les funcions han d'estar comentades per facilitar la lectura del codi.
- La carpeta `exe` ha de contenir l'executable generat per a les aules IA. En cas que el fitxer llegeixi algun fitxer de configuració, també cal incloure'l. No cal incloure cap imatge ni base de dades.
- La carpeta `doc` ha de contenir un petit PDF que expliqui els següents punts:

    1. La forma de compilar el codi (amb make, amb QtCreator, ...)

```
<?xml version="1.0"?>
<opencv_storage>
<imageName>image.jpg</imageName>
<hist_h type_id="opencv-matrix">
  <rows>50</rows>
  <cols>1</cols>
  <dt>f</dt>
  <data>
    1.01462454e-02 7.69015402e-02 4.73164916e-01 9.99999940e-01
    5.24805784e-01 4.94227111e-01 1.03981525e-01 8.95668566e-03
    3.70163023e-02 2.93891248e-03 4.75823926e-03 2.93891248e-03
    ....
    1.39948213e-04 1.39948213e-04 9.09663388e-04 6.99741067e-05
    2.09922320e-04 4.89818747e-04 0. 4.89818747e-04 1.39948213e-04
    2.93891248e-03 6.99741067e-05 4.19844640e-04 9.09663388e-04
    6.29766961e-04</data></hist_h>
<hist_s type_id="opencv-matrix">
  <rows>50</rows>
  <cols>1</cols>
  <dt>f</dt>
  <data>
    2.76091635e-01 4.67985123e-02 2.60426868e-02 8.14568251e-02
    1.12198941e-01 2.61014283e-01 5.49833536e-01 8.14568222e-01
    9.99999940e-01 7.79518306e-01 2.39083603e-01 1.03191696e-01
    ....
    5.22811823e-02 4.34697457e-02 2.40845904e-02 2.89798304e-02
    3.87703143e-02 3.64205986e-02 3.34834531e-02 2.80007832e-02
    1.37066767e-02 8.22400674e-03 3.13295494e-03 2.74133519e-03
    4.89524193e-03 2.93714483e-03 1.17485807e-03 3.91619280e-04 0.
    1.68396309e-02</data></hist_s>
<hist_v type_id="opencv-matrix">
  <rows>100</rows>
  <cols>1</cols>
  <dt>f</dt>
  <data>
    3.45092034e-03 5.75153390e-03 6.51840493e-03 1.34202447e-02
    5.75153390e-03 1.07361954e-02 1.22699384e-02 1.45705510e-02
    4.98466240e-03 4.60122712e-03 7.66871148e-04 1.15030666e-03 0.
    3.83435632e-04 3.83435632e-04 1.53374241e-03 1.91717793e-03
    ....
    2.53067482e-02 4.63957042e-02 3.41257676e-02 4.63957042e-02
    2.64570545e-02 4.52453978e-02 4.63957042e-02 2.95245387e-02
    4.48619649e-02 3.83435600e-02 7.17024505e-02 7.28527606e-02
    1.15414105e-01 1.34969324e-01</data></hist_v>
</opencv_storage>
```

Figura 1: Exemple de l'estructura XML d'un histograma. Per motius d'espai no es mostra el contingut complet dels histogrames.

2. Les parts que s'han paral·lelitzat sense entrar en els detalls de les funcions implementades. Per a les parts paral·lelitzades indicar quin tipus de construcció (bucle, tasca, seccions, ...) s'ha fet servir i raonar per què.
3. Exemples d'execució (amb temps d'execució).

La data límit d'entrega és el **17 de maig del 201**5. La part del codi computa un 60% de la nota, mentre que el document un 40%.